Implementation exercises for the course

Heuristic Optimization

Dr. Christian Camacho-Villalón and Dr. Thomas Stützle
Université Libre de Bruxelles — 2024

Implementation exercise sheet 1

Implement iterative improvement algorithms for the linear ordering problem (LOP). Information on the LOP is provided in the accompanying introduction to the implementation exercise. Apply the implemented algorithms to all the instances that are provided on TEAMS in the channel called "Implementation exercise 1". Also on this channel, there is a basic `C++` code that allows you to (i) read an instance, (ii) compute the objective function value, and (iii) generate random permutations.

The deadline for the implementation exercise is: **April 7, 2024 (23:59)**

Exercise 1.1

Implement iterative improvement algorithms with a

- first-improvement and another with a
- best-improvement

pivoting rule for each of the three neighborhoods transpose, exchange, and insert. For each of these three neighborhoods implement incremental updates (i.e., speed-ups) of the evaluation function value.

As a starting solution for iterative improvement, consider two possibilities. The first is to generate a random permutation, that is, to use the method "Uninformed Random Picking" — see the slides of lectures. The second is to use the heuristic of Chenery and Watanabe (CW) — see the slides of the lecture for details.

1. Apply the six resulting iterative improvement algorithms (all combinations of the two pivoting rules and the three neighborhoods) once to each instance. Do these experiments once using a random initial solution and once using the initial solution obtained with the CW heuristic (note that this results in 12 different combinations of starting heuristic and iterative improvement algorithm). Compute the following statistics for each of the 12 algorithms:
   - average percentage deviation from best known solutions;
   - total computation time across all instances.
2. Determine by means of statistical tests (in this case, the Student t-test or the Wilcoxon test), whether there is a statistically significant difference between the solutions generated by the different perturbative local search algorithms.
   Note: For applying the statistical test, the `R` statistics software can be used. The system is downloadable from `http://www.r-project.org/`. A short introduction to the most important commands for executing the tests was given in the introductory lecture to the implementation exercise.

Exercise 1.2

Implement a variable neighborhood descent (VND) algorithm. In this algorithm, consider the two possible (reasonable) orderings of the neighborhood relations:

- transpose, exchange, insert
- transpose, insert, exchange

Implement the VND algorithms only for the iterative first-improvement algorithms.

1. Compute the following statistics for each algorithm using as initial solution the heuristic of Chenery and Watanabe.
   - average percentage deviation from the best known solutions;

- total computation time across all instances.

2. Apply again the statistical tests to compare the solution quality reached by the two VND algorithms.

Additional information:

- In order to pass the exam, you have to successfully complete this implementation exercise.
- You need to do the implementation exercise by yourself — cooperation is forbidden.
- You have to submit the following items in **a zip folder with your name** via TEAMS:
  (i) a report in `pdf` format with scientific article structure (see examples provided on TEAMS) that concisely explains the implementation, reports the above mentioned statistical data (averages, standard deviations, and results of statistical tests), and interprets concisely the observed results;
  (Note: some of the criteria to be evaluated in the report are: the use of a scientific article structure (including abstract, introduction, problem description, material and methods, results, conclusion), the use of tables to present the obtained results, and the use of statistical tests to draw conclusion about the algorithms performance.)
  (ii) the source code of the implementation together with a `README` file explaining how to compile and/or execute the algorithms from a command line in Linux/MacOS;
  (Note: some of the criteria to be evaluated in the code are: compilation/execution without errors, the use of structures, code efficiency, indentation and comments.)
  (iii) a simple `.txt` file with the raw data that was used for statistical testing.

  It is important to stress that your code should compile/run on Linux/MacOS without errors and produce the requested outputs when executed on the provided instances; otherwise the implementation exercise will be considered insufficient.
- As programming language, you may use preferably `C`, `C++`, or `Java`. While using `Python` for this exercise is also possible, we recommend against it because it is much more slower than the alternatives. The sample routines are available only in `C++`, but you are free to adapt them at your convenience. Please make sure that your code is properly commented and indented, and that the `README` file mentions the exact commands for its compilation and execution.
- The algorithms implemented in this exercise will be re-used in the second implementation exercise.

- **Happy coding :-D**